Enterprise-Component Models

By :

Eric LEFEBVRE,

Ghislain LÉVESQUE

Topics :

- Object-Oriented enterprise modelling
- Business analysis patterns
- Enterprise-wide Information System Architecture : methods and techniques to build it rapidly and accurately
- Reusability of enterprise-components

ABSTRACT

Enterprise-wide Information System Architecture has been considered for a few years to influence the success of the MIS function.

Meanwhile, the emerging Object Oriented (OO) techniques hold methods that will allow to :
1) develop faster and
2) distribute systems across internet and intranet networks.

Today, component-based development (CBD), a natural evolution of OO development, appears to be the best development approach, mainly because of its capacity for reusability and, therefore, its potential for saving time and effort. The market for CBD tools and frameworks is rising at 80% per year.

Initially based on data and function models, the architecture has to evolve towards OO techniques, thus expanding its role to serve as a foundation for component-based development.

In this paper, we present an original and particularly effective method to build such an Enterprise-wide Information System architecture. The method encompasses:

- adaptation of a generic process model, based on Porter value chain, to the enterprise;
- design of Enterprise-Component Models by identifying actors, roles and objects involved in processes ;
- creation of a component library from these models.

We describe how to build the enterprise-component models, starting from a business process model and representing it in a generic pattern; this pattern is then used at a higher level to identify and represent enterprise components. The method is illustrated step by step to get the high-level model and an example of a typical enterprise-component is given.

Such a method offers many advantages :

- system consistency and reusability of components ;
- integration of processes and systems that support them ;
- rapid development of new components when applying these models to other processes.

A component library, the most important result of the method, can then be expanded in several ways, as more and more components become available from software or application vendors.

About the authors

Éric Lefebvre is the director of research at PROGESTIC Group, an IT consulting firm. He holds a Ph.D of l'Université de Grenoble. He has done intensive research work on generic elements of the entreprise IS models. He is a research associate of the « Laboratoire de Recherche en Gestion des Logiciels de l'Université du Québec à Montréal ».

Ghislain Lévesque is a professor at l'Université du Québec à Montréal. He holds a Ph.D of l'Université Panthéon-Assas (Paris II). He is the author of a recent book on methods and techniques of OO analysis and design. He is a research associate of the « Laboratoire de Recherche en Gestion des Logiciels de l'Université du Québec à Montréal ».

Note

Component Models presented in this article result from a work being done by Éric Lefebvre and Peter Coad, in order to write a book entitled « Enterprise-Component Models », to be published by Prentice-Hall, in spring 1999.

Introduction

Information systems development through the assembly of reusable building blocks has become reality today.

John Zachman (1982) had a clear vision when he stated that Information Systems would be more and more developed from pre-defined elements. Since that time, methods, techniques, and tools have clearly evolved following this direction.

Zachman's vision was confirmed by Meta Group, a well-known Information Technolgy (IT) market assessment and consulting firm. Its analysts announced that development by component was an important trend and that both the integrated enterprise systems and the development tools vendors should follow it (Meta Group, 1997).

Products and tools available today reflect this trend. Many products claim the use of components, patterns, frameworks, blueprints, and templates, that can all be classified as pre-defined reusable elements. For instance, development environments for Object Oriented (OO) programming languages such as Java are sold today with component libraries that are continuously expanding.

The two examples below of the Enterprise Resource Planning (ERP) systems and of the IBM SanFrancisco project also demonstrate this trend.

The ERP systems are enterprise integrated systems that have been adopted by a large number of companies. They provide generic components that can be reused and adapted by most enterprises. SAP, the dominant ERP systems vendor, recommands to start the implementation by adapting its generic business blueprint. The SAP business blueprint (Curran and Keller, 1998), is a set of models representing the enterprise events and activities. Systems developers can tailor this blueprint to their enterprise, by selecting the events and activities that apply. The business applications are then generated according to the blueprint.

The other example is provided by the IBM SanFrancisco project (Bohrer et al, 1998). SanFrancisco provides a framework for application developers to build on when producing enterprise applications. SanFrancisco has a layered architecture, with low-level infrastructure and service components and mid-level common business objects, together with high-level industry-specific core business processes. This framework is supposed to assist the developers in developing efficient Java applications.

The technological environments of the vendors of ERP systems and those of SanFrancisco framework tend to introduce some limits to their customers use and consequently rule out some of their intended benefits. Furthermore, their installation at customer site and their maintenance require major investments. On the contrary, the proposed set of Enterprise-Component models (ECM's) presented in this article is intended for an easy adaptation to business and technical requirements of any enterprise.

This article presents first the importance of the Enterprise Wide Information Systems Architecture (EWISA) and the growing interest in current OO techniques for system development. This analysis concludes that an EWISA should be built using OO techniques.

Then, a domain-neutral component is defined, from the generic elements of a business process. A high-level enterprise model is built and ECM's are derived from this model.

The article concludes by describing future efforts to detail and improve these models and to define the process to reuse them.

1. Enterprise-wide Information System Architecture

EWISA is considered one of the critical success factors of the MIS function, since it provides the business blueprint required to integrate the main enterprise information systems. This architecture is composed of enterprise function and data models that allow to identify the main enterprise applications and relationships between them. Several research works have proven the importance of EWISA. For instance, Lefebvre (1996) proved that IS planning success was significantly higher for the companies that had such an architecture.

IBM Business System Planning (1981) and Information Engineering (James Martin, 1982 and 1990) are the main methods based upon an EWISA that have inspired the system planning and development practices. These methods can be greatly enhanced by the use of generic elements.

A group of IBM analysts developed, for their internal use, a technique to build an EWISA, called Business Information Analysis and Integration Technique (BIAIT, Kerner, 1979), which was based on generic elements and on a set of eight questions that helped to adapt them to any context.

Some business domains are generic : the resource management domains (material, human, and financial) and the facility management domain (i.e. the infrastructure required to produce and deliver the goods to customers). For these domains, functions and data are the same for any enterprise.

The production domain depends on the enterprise, since the production activities, ranging from the customer's request to the delivery of goods, will depend on the type of products and services. The eight BIAIT questions apply to this domain and the binary answers allow to adapt the generic elements to the enterprise. One of these eight questions is, for instance :

« Does the enterprise sell from stocks? »
If the answer is « yes », then stock management functions and data will be required.

Porter (1985) defined a high-level enterprise process model, with the objective to identify the value chain of an industry, that is to say the processes that brought a direct value to the delivery of products or services to the customers. This model was also based on generic elements and was frequently reused to build EWISA (Martin, 1995, Gale and Eldred, 1996). It identified the enterprise primary activities that brought value to the goods and the activities that support the primary ones.

Lefebvre (1993) evaluated a generic enterprise IS model, based on previous models and techniques. He experimented significant savings in its use. With PROGESTIC Group (a Montreal-based IT consulting firm), they further developed a tool that assists the architecture building process, thus significantly increasing its efficiency. This made it therefore possible to build an EWISA during the planning process, without taking too much time and effort.

However this architecture was still based on function and data models.

2. Object Oriented techniques

OO techniques have existed for many years, but their acceptance is recent and largely due to the rapid Internet growth. Potential benefits of OO techniques are in :

- developing faster and
- distributing systems across internet and intranet networks.

An advantage for the developers is that the same model representation applies throughout the whole development cycle. Business objects are the first category of objects to identify. They are further detailed with their attributes (what they are), their methods (how they behave), and their relationships (who they know). Other categories of objects to be added next are : User Interface objects, data access objects, and objects to communicate with other systems.

The concept of a component, by analogy with the manufacturing industry, has recently been developed. A component is a type, class of objects or any other workproduct that has been specifically engineered to be reused (Jacobson and others, 1997). Component-based development (CBD) appears to be the best development approach, mainly because of its capacity for reusability and, therefore, its potential for saving time and effort. Market for CBD tools and frameworks is rising rapidly.

Programming languages have evolved accordingly. OO programming languages, such as C++ and, more recently, Java are becoming popular. Other popular languages, such as Visual Basic, can be used as OO languages.

System architectures have evolved from mainframe computers to 2-tier and, recently, to 3 and even n-tier Client/Server architecture. Web servers, for instance, bring an additional tier by allowing systems to be expanded through Internet.

Well accepted standards such as CORBA and DCOM rule the message exchanges between objects. Other well accepted standards SQL and XA rule the database access. This facilitates the data distribution. Furthermore, Relational Data Base Systems, such as Oracle version 8, have now the capability to manage objects and transaction monitor systems, such as CICS, support distributed processing.

As a net result, mission critical distributed systems can now be developed and connected to the Internet and enterprises can adopt OO techniques and methods for the whole life cycle of their applications. Therefore EWISA, initially based on data and function models, should evolve towards OO and component models, while taking advantage of well accepted generic elements.

3. Building the Entreprise-Component models

The method that has been used to build the ECM's composing an EWISA relies on the analysis of a business process. The elements of a business process defined through analysis have been included in a generic component called a « domain-neutral component », because it applies to any enterprise domain.

By using this component, a generic high-level enterprise model can be built and a set of ECM's covering most of the enterprise domains can be obtained. Here are the steps followed.

3.1 Business process description

The process model is the model that unifies the other OO analysis and design models (Lévesque, 98). Processes are also easier to identify than any other business entity, because they are more concrete. It is therefore recommended to start the analysis by identifying processes and their respecting events. The whole enterprise can be seen as a set of processes triggered by events.

Each process is composed of activities, each activity being triggered by an event. An event might be generated at a given time, if the activity has been scheduled, or might come from an actor (an actor can be a person or an organisation unit or an automated machine), who has expressed a request (for instance : a customer who sends an order).

An activity involves things, and requires things for its execution.

- Things involved in an activity are changing their state accordingly. A change of state is then generally recorded at the end of the activity;
- Things required to execute an activity are resources (human, material, or/and financial resources) or facilities (space or equipment).

The costs represented by the resources are charged to a project or a cost center account.

Things have a description, which is independent of their state.

An activity generally produces an output event, which may again trigger another activity.

3.2 Domain-neutral component

The business process, as described above, has to be converted into an Information System representation. The classes of objects that best describe processes have been conveniently classified in four categories :

Moment-interval
A moment-interval represents objects that are time-dependent, i.e. activities and events. Moment-intervals tie together a component model. They express the heart of what that component is all about. In a model, moment-intervals embody important methods to calculate results and to evaluate them.

Role
A role might be fulfilled by a person, an organization unit or a machine. It generates an event or accomplishes an activity. It is associated to a moment-interval.

Roles include methods to assess actor's performance.

Thing
Things are the next objects to be identified. They are :

- things involved in the production activities as products and work-in-progress;

- things involved in maintenance activities as the facilities (building, equipments or tools);
- things required by the activities as resources (human, material or financial) or facilities.

Description
Descriptions are the last part. They include static data and methods that collect data, such as available quantities. Things may have corresponding descriptions.

In practice (Coad and Lefebvre, 98), it was found very useful and efficient to represent these four categories by four different colors. There are two main reasons for it. First, it helps to identify classes of objects and, therefore, to build models. Second, it helps to read models.

Color assignments are the following :
- pink for moment-intervals;
- yellow for roles;
- green for things;
- blue for descriptions.

This is shown in figure 1 (models presented in this article have been built with Together, a Case tool from Object International).

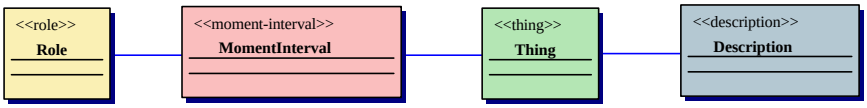

Figure 1. The four categories and their assigned colors.

Each of these four categories corresponds to stereotypical responsibilities, the kind of behavior that one would expect such a class to exhibit.

These stereotypical responsibilities include attributes and links (Figure 2):

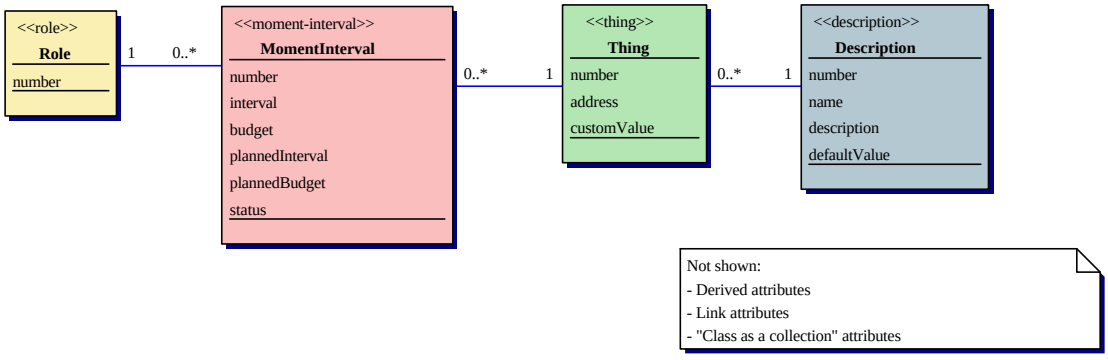

Figure 2. The four categories and their stereotypical attributes and links.

...methods (Figure 3):

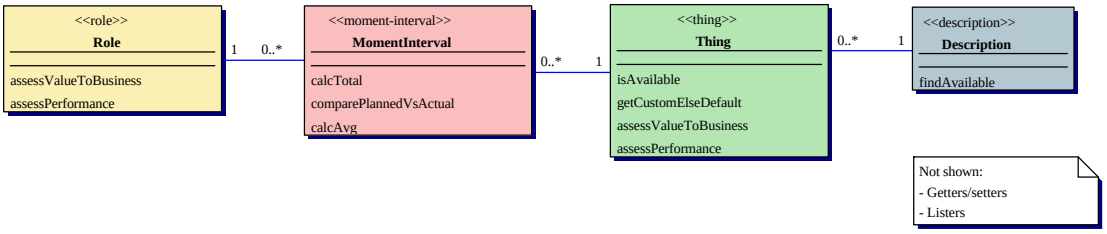

Figure 3. The four categories and their methods.

...and pairs of plug-in points and default plug-ins (Figure 4):

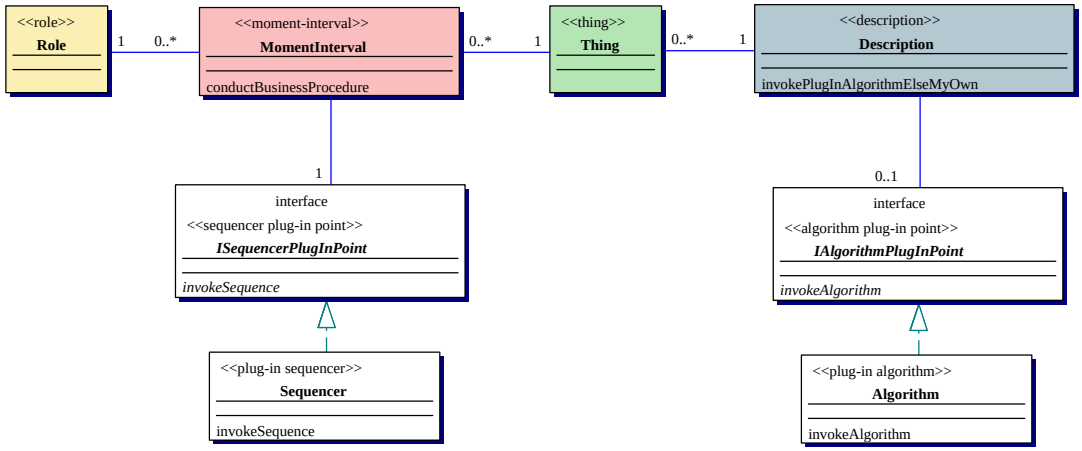

Figure 4. The four categories and their stereotypical plug-in points and default plug-ins.

Diagrams are completed by:

- Notes, especially for features lists
- Interfaces, especially for plug-in points and plug-ins
- External system classes, those classes that represent external systems that the system being developed must interact with.

The entire domain-neutral component is shown in Figure 5.

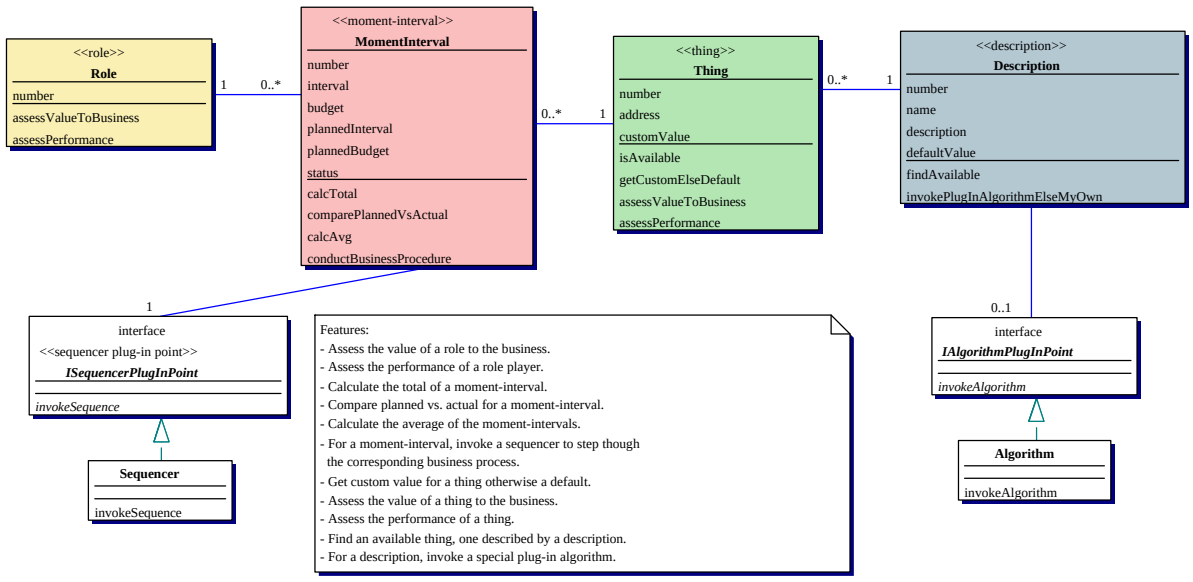

Figure 5. A domain-neutral component.

This domain-neutral component can be reused to model any enterprise domain.

3.3 High-level enterprise model

When reusing the domain-neutral component to build a high-level enterprise model, the main enterprise classes of objects are first identified. This method consists of the following steps :

- identifying the moment-intervals, by pointing out the main activities and events of an enterprise (the pink classes);
- adding the roles that generate the events and execute the activities (the yellow classes);
- adding the things on which the activities are executed and that are required for the execution (the green classes);
- adding the descriptions of the things (the blue classes).

In the high-level model, the key categories of the domain-neutral component will be reused with the following meaning :

Moment-Intervals
Production activities are the first moment-intervals to consider, when an entreprise model is built. Each moment-interval :

- participates in the direct flow of production of a product;
- covers the product life cycle, from order to delivery;
- depends on the products of the enterprise (they will be later decomposed in sub-domains according to this dependency).

The main events to consider are the **sale orders** from the customers (customers are the reason of the enterprise to exist) and the **purchase orders** to the suppliers (to get the facilities and the external resources that are required).

Roles
The first roles in the enterprise are :

- **General management** (or owners) who states the objectives and selects the products to sell;
- **Customer** who issues a request to buy;
- **Supplier** who answers a purchase request.

Things
The most important things are the **products** sold to customers, that are the results of the production activities.

The next ones are :

- Resources required to execute the production activities (**Human Resources** and **Material Resources**);
- **Facilities** to use;
- **Accounts** to post the accounting transactions (sales, purchases and resource consumption).

Descriptions
The main generic enterprise catalogues are the following :

- **Catalogue of products** sold to customers;
- **Supplier catalogue of products** purchased from suppliers;
- **Organisation chart**, that describes the hierarchy of organisation units;
- **Chart of accounts**, which rules the way to report to the management.

Once the classes have been identified, the model is completed by defining the relationships between them and the attributes and methods of each class, adapting the domain-neutral component with domain specific features.

The resulting model (attributes and methods have been kept hidden to simplify the model) is shown in Figure 6.

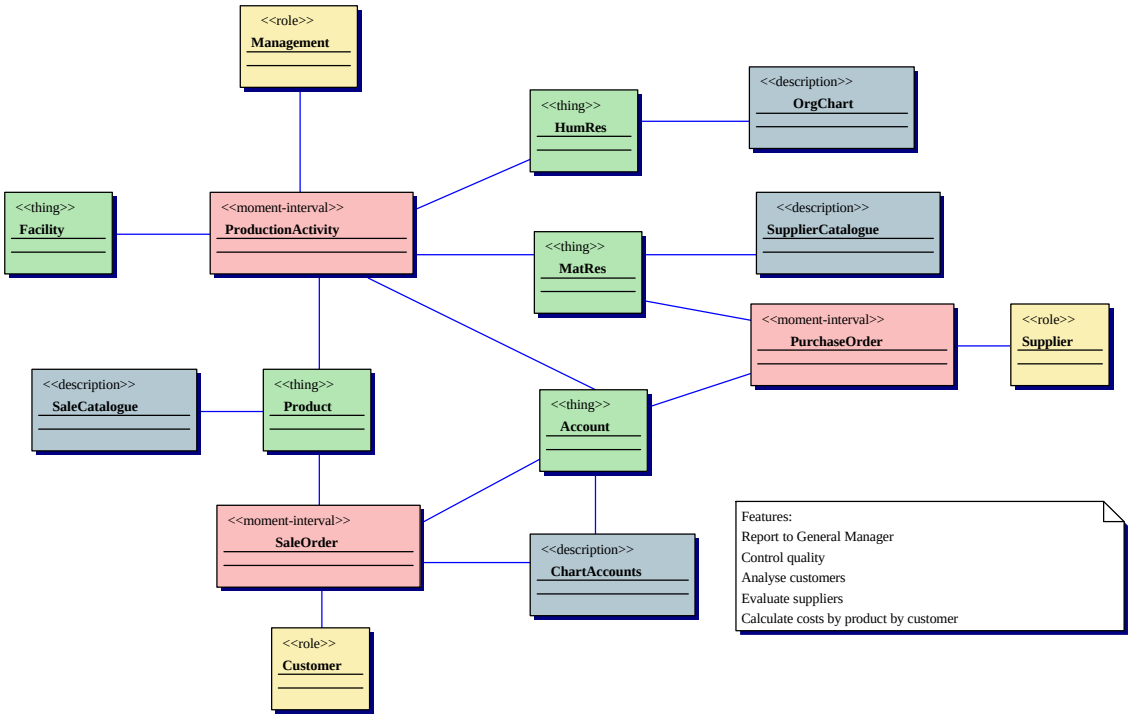

Figure 6. The high-level enterprise model.

This model reveals interesting analogies that will help to identify patterns, as shown in the tables below :

- analogies between the customer chain and the supplier chain;
- analogies between Human Resource and Material Resource management required by the activities.

	Moment-interval	**Role**	**Thing**	**Description**
Customer chain	SaleOrder	Customer	Product	ProductCatalogue
Supplier chain	PurchaseOrder	Supplier	MaterialResource	Suppl.Catalogue

	Moment-interval	**Role**	**Thing**	**Relationship**
MaterialResource	User request	Manager	Employee	Cost accounting
HumanResource	User request	Manager	MaterialResource	Cost accounting

The high-level enterprise model serves as a base for building detailed ECM's, as described in the next section.

3.4. Enterprise-Component models

The high-level enterprise model can be specified by a comprehensive set of ECM's.

Production activities, the primary enterprise activities, can be decomposed into :

- manage orders (if orders are taken)
- negotiate specifications (if each product is specified at order)
- make product (if product is made by the enterprise)
- manage stock (if product is sold from stock or if material resources are stocked))
- bill customer or cash
- manage renting (if product/service is rented)
- record sales history (if sales data are stored for analysis)
- manage clientele inventory (if the sold product has to be tracked)

These activities exist in a given enterprise depending on the types of products that are delivered. They can be included in a global enterprise model, according to the answers to the BIAIT questions, mentioned in parentheses.

Events can also be decomposed. For instance, the following events occur to fulfill a customer order :
Customer order

- Order request
- Request for proposal
- Delivery
- Invoice

Referring to the strategic triangle (Levesque and Lefebvre 1998), each thing of the high-level enterprise model related to production activities has a counterpart in the resource management domain of each enterprise corresponding to the topic of a category of enterprise-component models, such as :

- Facility (or fixed assets)
- Human resource
- Material resource
- Account and financial resource

The table below presents the ECM's of our EWISA.

Category	Components
Order management	
	Order request
	Request for proposal
	Delivery
	Invoice
	Sales analysis
Manufacturing management	
	Specifications
	Production planning
	Production
	Quality control
Facility management	
	Acquisition and installation
	Maintenance
Activity management	
	Planning
	Execution
Material resource management	
	Request MR
	Supplier selection
	Request satisfaction
	Use MR
Human resource management	
	Enterprise organization
	HR Request
	HR Assignment
	HR Pay
	HR Development
Budget and account	
	Sales Accounting
	Purchase Accounting
	Budget
	General Accounting
	Cost Accounting

As an example of an ECM the Order Request Model of the Order Management category is presented. The object classes are first identified in the table below.

	Moment-interval	Role	Thing	Description
Order request	Customer Order	Customer Sales representative	Product Ordered	Cust.Catalogue

The model is completed with the attributes and methods from the domain-neutral component, adapted to take into account the domain-specific features. It is presented in the figure 7.

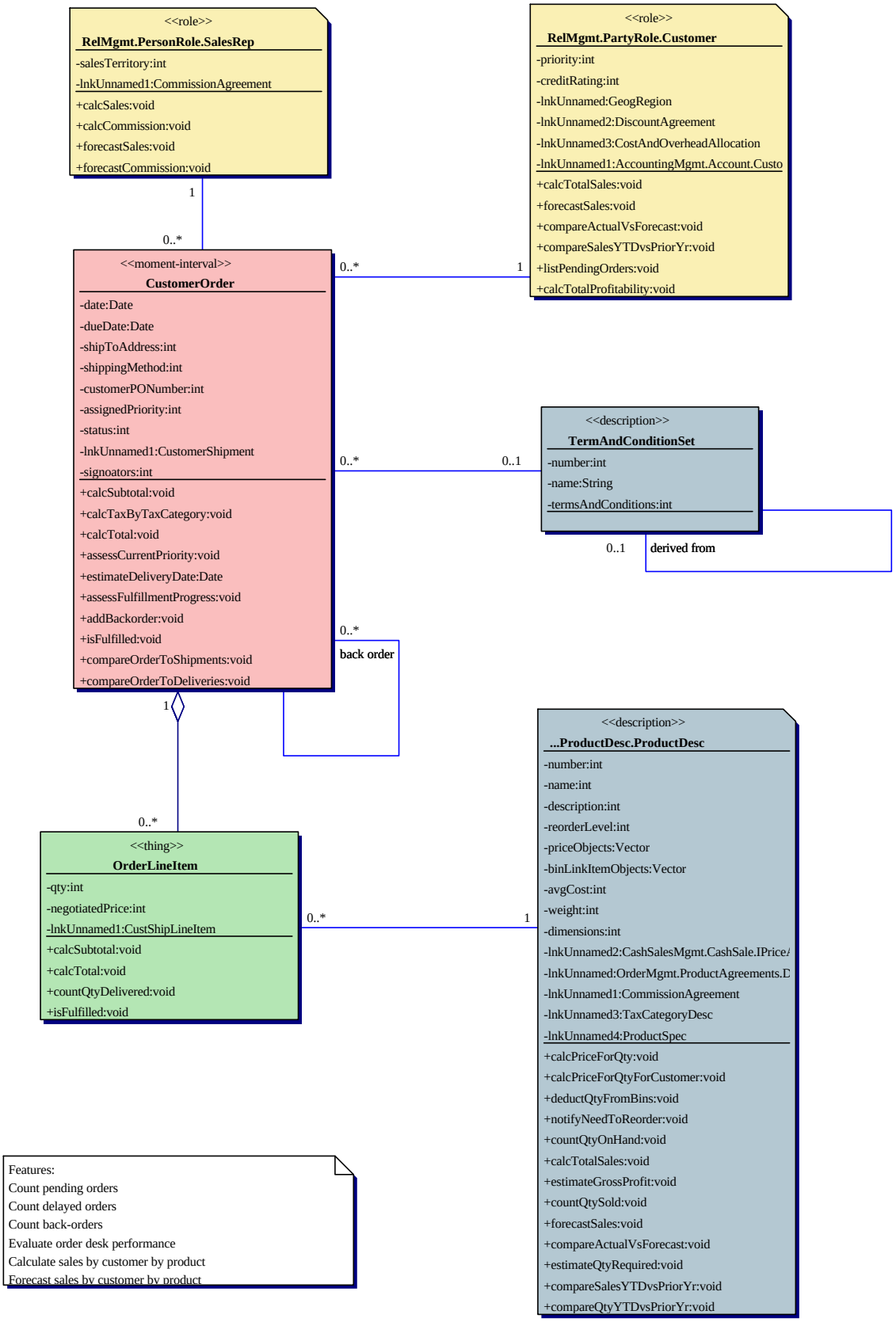

Figure 7. A Customer Order component model.

4. Future developments

EWISA built on the set of enterprise-components defined in our models with respect to OO techniques brings the following advantages :

- insuring system consistency and integration,
- providing components that are reusable, extensible, and customizable with plug-ins.

The models engender larger-scale starting points for building better models, moving beyond mere class-at-a-time analysis and design techniques.

Our current effort is aimed at defining the process to reuse the components.

The ECM's could also be expanded in several ways :

- more details can be added to the existing models, for instance new classes, new attributes, new methods;
- new components can also be added;
- parameters can be defined to easily adapt these models to different contexts.

This expansion will result from various experiences gained in using the ECM's to build EWISA's.

New tools should more and more assist the developers to reuse the components, to modify them, and to integrate them with components from other sources, such as service components.

Finally, we think that more and more business as well as service components will become available from both software and application suppliers, thus reenforcing Zachman's vision of system development by building blocks.

References

Bohrer K. and al., «Business Process Components for Distributed Object Applications», Communication of the ACM, Vol.41, No.6, June 1998, pp. 43-48.

Coad, P., and Lefebvre, É., « Enterprise-Component Models in color », Coad-letter, www.oi.com, July 1998.

Curran, T., and Keller, G., « SAP R/3 Business Blueprint », Prentice Hall, 1998.

Gale, T., and Eldred, J., « Getting Results with the Object-Oriented Enterprise Model », SIGS, New-York, 1996.

IBM Corporation, « Business System Planning », Information Systems Planning Guide, Publication No. GE20-0527, 1981.

Jacobson, I., « Software Reuse », Addison-Wesley, New-York, 1997.

Kerner, D.V., « Business Information Control Study », Data Base 10, No4, Spring 1979, pp.10-17.

Lefebvre, É., « Évaluation d'un modèle générique pour la construction de l'architecture des systèmes d'information de l'entreprise », Mémoire de maîtrise, UQAM, 1993.

Lefebvre, É., « Améliorer les méthodes de planification informatique : une approche pluraliste », Thèse de doctorat, Université Pierre Mendès-France-Grenoble II, 1996.

Lévesque, G., « Analyse de système orientée-objet et génie logiciel », Chenelière/McGraw-Hill, Montréal, 1998.

Lévesque, G. et Lefebvre, É., « Une architecture des SI fondée sur les objets de métier de l'entreprise: un atout stratégique », 1er symposium international sur l'évaluation stratégique, Université Paris II, 6 et 7 juillet 1998.

Martin, J., « Strategic Data Planning Methodologies », Prentice-Hall Inc., Englewood Cliffs, NJ, 1982.

Martin, J., « Information Engineering », Prentice-Hall Inc., Englewood Cliffs, NJ, 1990.

Martin, J., and Oddell, J., « Object-Oriented Methods : a Foundation », Prentice-Hall Inc., Englewood Cliffs, NJ, 1995.

Meta Group, « The Webolution of Business Intelligence », text of a conference, Montreal, May 1997.

Porter, M.E., « Competitive Advantage : Creating and Sustaining Superior Performance », New-York, Free Press, 1985.

Zachman, J., « Business Systems Planning and Business Information Control Study : a Comparison », IBM Systems Journal, Vol.26, No3, 1982, pp.276-292.